transform 转换

1 translate 平移

1. 数值：允许出现负值

- 一个值: translate(100px) 在x轴上移动100px ；
- 两个值: translate(100px,200px) 在x轴上移动100px y轴上移动200px

2. 百分比: 基于当前盒子的宽高

- translate(50%) 在x轴上移动自身宽度的50%；
- translate(50%,50%) 在x轴上移动自身宽度的50%，在y轴上移动自身高度的50%

3. translateX() 只在x轴上进行平移

- transform: translateX(200px);

4. translateY() 只在y轴上进行平移

translate 最大的优点：不会影响到其他元素的位置

translate 中的百分比单位是相对于自身元素的

对行内标签没有效果

2. 水平垂直双方向居中

1. 绝对定位的方式

position: absolute;

top: 0;

left: 0;

bottom: 0;

right: 0;

margin: auto;

2. transform的方式

/* transform变形平移的方式 */

position: absolute;

left: 50%;

top: 50%;

transform: translate(-50%,-50%);

向左移动自身宽度的一半,向上移动自身高度的一半

```
<style>
    * {
     margin: 0;
     padding: 0;
```

```
    }

      div {
      /* 配合定位 */
      position: absolute;
      top:50%;
      left: 50%;
      /* 向左移动自身宽度的一半,向上移动自身高度的一半 */
      transform: translate(-50%,-50%);
      height:  273px;
      width: 273px;
      background-color: pink;
    }
</style>
```

3. 旋转 rotate

1. 2D旋转指让元素在2维平面内顺时针旋转或者逆时针旋转

- transform: rotate(度数)
- rotate 里面跟度数，单位是 deg 比如 rotate(45deg)
- 角度为正值，顺时针，负值，为逆时针
- 默认旋转的中心点是元素的中心点

```
<style>
        * {
            margin: 0;
            padding: 0;
        }

        .demo {
            height: 300px;
            width: 300px;
            background: url(./img/1.jpg) no-repeat 0px 0px / cover;
            /*
            rotate  旋转
            单位:  deg
            正值:  顺时针旋转
            负值:  逆时针旋转
            */
```

```
            transform: rotate(0deg);
        }
    </style>
```

4. 缩放 scalex

对元素进行缩放的函数

- scalex() 水平方向缩放
- scaleY() 垂直方向缩放
- scale() 双方向的缩放
- >1的值表示 放大多少倍
- <1的值表示 缩小多少倍

注意其中的x和y用逗号分割:

- transform:scale(1,1): 宽和高都放大一倍，相当于没有放大
- transform:scale(3,3)：宽和高都放大了3倍
- transform:scale(3)：只写一个参数，第二个参数则和第一个参数一样，相当于 scale(3,3)
- transform:scale(0.5,0.5)：缩小
- sacle缩放最大的优势：可以设置转换中心点缩放，默认以中心点缩放的，而且不影响其他盒子

```
<style>
        * {
            margin: 0;
            padding: 0;
        }

        .demo {
            height: 300px;
            width: 300px;
            background: url(./img/1.jpg) no-repeat 0px 0px / cover;

            /*
            scale 缩放
            >1的值表示  放大多少倍
            <1的值表示  缩小多少倍

            scale(1,2)  表示x轴放大1倍,y轴放大两倍
            */
            /* transform: scale(1,2); */
```

```
        }

        .fa {
            /* 浏览器能够识别的最小字体就是12px */
           font-size: 12px;
           /*
           面试题:
           如果想让字体在12px以下,就可以使用scale
           */
           transform: scale(0.5);
        }
    </style>
```

5. 缩放隐藏

给超出的部分设置隐藏（overflow: hidden;）

```
<style>
        * {
            margin: 0;
            padding: 0;
        }

        div {
            width: 550px;
            height: 550px;
            border: 2px solid  red;
            margin:200px auto;
            /*
            超出部分隐藏:为了让滑过变大的图片不超出div这个盒子
             */
            overflow: hidden;
        }

        div:hover>img {
            transform: scale(1.1);
        }
    </style>
```

6. skew:倾斜

- skew(0deg) 表示在x轴上进行倾斜
- skew(0deg,0deg) 第一个值表示在x轴上进行倾斜,第二个值表示在y轴上倾斜
- skewX(0deg) 只在x轴上倾斜
- skewY(0deg) 只在y轴上倾斜

```
<style>
        * {
            margin: 0;
            padding: 0;
        }

        .demo {
            height: 300px;
            width: 300px;
            background: url(./img/1.jpg) no-repeat 0px 0px / cover;
            /*
            skew:倾斜
            skew(0deg) 表示在x轴上进行倾斜
            skew(0deg,0deg) 第一个值表示在x轴上进行倾斜,第二个值表示在y轴上倾斜
            skewX(0deg)    只在x轴上倾斜
            skewY(0deg)    只在y轴上倾斜
            */
            transform: skewY(0deg);
        }
    </style>
```

7 透视perspective

1. 透视：在2D平面产生近大远小视觉效果
- 透视的单位是像素
- 透视写在被观察元素的父盒子上

2. 必须给父级增加视距 perspective
- transform: translateZ(200px);
- translform:translateZ(200px)：仅仅是在Z轴上移动。

有了透视，就能看到translateZ 引起的变化了

- translateZ：近大远小
- translateZ：往外是正值
- translateZ：往里是负值

3. 组合写法: translate3d(x,y,z)

- transform: translate3d(0px,0px,0px);

```
<style>
        * {
            margin: 0;
            padding: 0;
        }
        body {
            /* 视距:近大远小的视觉效果 */
            perspective: 800px;
        }

        .demo {
            height: 200px;
            width: 200px;
            margin:200px auto;
            background-color: red;
            /* 必须给父级增加视距 perspective*/
            transform: translateZ(200px);
            /*
            组合写法:
            translate3d(x,y,z)
            */
            transform: translate3d(0px,0px,0px);
        }
    </style>
```

8 旋转组合

- rotateX(0deg) 绕x轴进行旋转
- rotateY(0deg) 绕y轴进行旋转
- rotateZ(0deg) 绕Z轴进行旋转

组合写法

- transform:rotate3d(0,0,0,0deg)
- 前三个表示x,y,z轴的矢量,控制x,y,z轴是否发生旋转
- 0表示不发生旋转,1表示发生旋转

```
<style>
        * {
            margin: 0;
            padding: 0;
        }

        .demo {
            height: 300px;
            width: 300px;
            background: url(./img/1.jpg) no-repeat 0px 0px / cover;
            /*
            rotateX(0deg)  绕x轴进行旋转
            rotateY(0deg)  绕y轴进行旋转
            rotateZ(0deg)  绕Z轴进行旋转

            组合写法
            transform:rotate3d(0,0,0,0deg)
            前三个表示x,y,z轴的矢量,控制x,y,z轴是否发生旋转
            0表示不发生旋转,1表示发生旋转

            */
            /* transform: rotateZ(0deg); */
            /* 只开启x轴 */
            /* transform: rotate3d(1,0,0,0deg); */
            /* 只开启y轴 */
            /* transform: rotate3d(0,1,0,0deg); */
            /* 只开启z轴 */
            /* transform: rotate3d(0,0,1,0deg); */
            /* 同时开启xyz轴 */
            transform: rotate3d(1, 1, 1, 0deg);
        }
    </style>
```

9 3D呈现transform-style

- 3D呈现：transform-style
- 让子元素开启三维立体环境
- transform-style: flat 子元素不开启3d立体空间 默认的
- transform-style: preserve-3d 子元素开启立体空间

- 代码写给父级元素，影响子盒子

```
* {
    margin: 0;
    padding: 0;
}

.fa {
     height: 400px;
     width: 400px;
     /* border: 10px solid red; */
     margin: 200px auto;
    /* 3d效果 */
     transform-style: preserve-3d;

     transform: rotate3d(1,0,1,45deg);
     animation: move 5s linear infinite;
}
```